電腦輔助設計及製造
(CAD/CAM)

陳至剛 編著

第4章

微電腦APPLE II 繪圖的原理與應用

4-1 APPLE II 繪圖的基本原理

APPLE II 是以掃描線來顯像，也就是說我們可用一些掃描線（水平線）來組成顯示幕，每條掃描線再以掃描點（pixel）構成。APPLE II 的高解析度顯示模式共有 192 條掃描線，每條線包括 280 個掃描點。我們將掃描線編號爲 0 到 191；每條線中的掃描點編號爲 0 到 279。因此每個掃描點可由兩個數來決定：水平線的號碼和它在線中的號碼。掃描點的發亮與否直接決定在顯示幕上產生影像。

繪圖語言必須具備顯現個別掃描的能力，在 Applesoft 中，HPLOT 指令即執行這個動作。HPLOT X,Y 可使 Y 掃描線上第 X 位置的掃描點發亮。例如 HPLOT 0,0 繪出顯示幕左上角的點，HPLOT 279,191 繪出右下角的點。採用這個方法，我們可以很簡單地控制所有掃描線上任意位置掃描點的發亮與否。

我們要畫的影像大多是由許多點構成。在第一個程式範例裏

我們只用HPLOT 指令來畫出影像。

＜例1＞　試以BASIC 程式繪出一個夜晚的星象出來

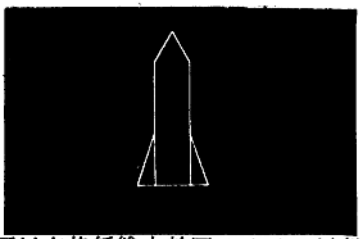
```
1  REM  PROGRAM    • (SPACE)
2  REM  PLOTS RANDOMLY SELECTED POINTS
10  HOME : HGR : HCOLOR= 3
20  FOR I = 1 TO 100
30 X =  RND (1) * 279:Y =  RND (1) * 191
40  HPLOT X,Y
50  NEXT I
```

則CRT上顯示如圖所示

在本程式中，30 決定一個隨機位置把「星星」畫出。每次執行這個程式都會產生新的「星象」。

＜例2＞　以BASIC 程式繪出一個太空船的圖形

```
1  REM  PROGRAM      (SPACE SHIP)
2  REM  USES HPLOT TO DRAW ELEMENTARY SHAPS
10  HOME : HGR2 : HCOLOR= 3
20  HPLOT 120,170 TO 120,50 TO 140,20 TO 160,50 TO 160,170
30  HPLOT 160,120 TO 180,170 TO 100,170 TO 120,120
```



本題我們亦可以方格紙繪之於圖4-1.1，以供讀者們參考比較之用。

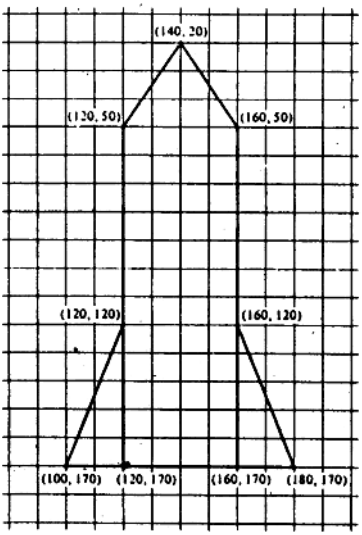
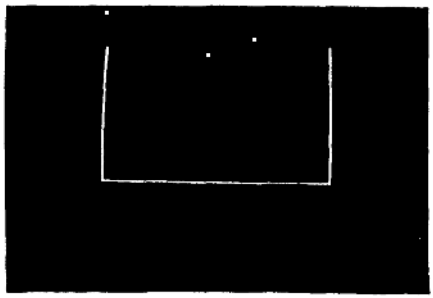


圖4-1.1　太空船之圖形

＜例 3 ＞　試寫出一個模擬的球跳躍的程式

```
1  REM  PROGRAM      (BOUNCING BALL)
2  REM  SIMULATES A BALL BOUNCING OFF FLOOR AND WALLS
10 DX = 4:X = 8:V = 0:A = 2:V = 0
20  HOME : HGR
30  HCOLOR= 2: HPLOT 2,0 TO 2,133 TO 266,133 TO 266,0
40  HCOLOR= 3: HPLOT X,Y: HPLOT X + 1,Y
50  IF Y = 132 THEN V =  - V: IF V = 0 THEN V =  - 20
60  IF X > 263 THEN DX =  - DX
70  IF X < 6 THEN DX =  - DX
80 V = V + A
90  HCOLOR= 0: HPLOT X,Y: HPLOT X + 1,Y
100 Y = Y + V:X = X + DX
110  GOTO 40
```

“球”由 40 列產生。爲了使球看得更清楚，我們畫了兩個並排的點。30 列繪出使球反彈的「牆壁」和「地板」。當球在跳動時，設定速度 DX 控制它的水平運動，以加速度 A 和變速度 V 控制它的垂直運動。我們藉著下面的方法模擬球的運動：先畫一個白點（ 40 列），然後在相同位置畫一個黑點來擦掉原來的點；改變 X ，Y的位置（ 100 列）而在新的位置再畫一個白點。50～70列控制球從牆壁和地板反彈的情形。爲了保持不斷的運動， 50 列在球反彈過低（ Y＝132 和V＝ 0 ）的時候將球加速



。在按下 CTRL-C 或RESET中斷程式之前，球將不斷地跳來跳去。

4-2　如何分配及應用APPLEII的記憶體

APPLE II 的圖形輸出實際上即是顯示一段記憶體的內容。由於這個特性，首先右圖 4-2.1中所表示的記憶乃是原來的標準。我們以十六進位（在字頭前加$符號）及十進位來表示記憶體位址。

APPLE II 有48 K byte 的 RAM（Random Access Memory，隨機存取記憶體），再加上16 K byte 的ROM（Read Only Memory, 唯讀記憶體）。K的值，表示1024，因此48 K爲48（1024）= 49152，而64 K等於64（1024）= 65536 。這些數目看起來很奇怪的原因是我們平常使用十進位而不是二進位數。但電腦是二進位的機器，所以65536 可用二進位表示成1000000000000000，對電腦來說並不是很複雜（除了它的長度之外）。

每個記憶單位[位元組（byte）]含有8位元（bit）的資料，每個位元可爲0或1。位元組在記憶體中的位置稱爲位址（address），可用十進位，二進位或十六進位來表示位址和它所含的資料，例如「5005 記憶體位置的內容爲65」，我們有三種方法來描述這個記憶單位：

0001001110001101 [01000001]	5005 [65]	138D [41]
二進位表示	十進位表示	十六進位表示

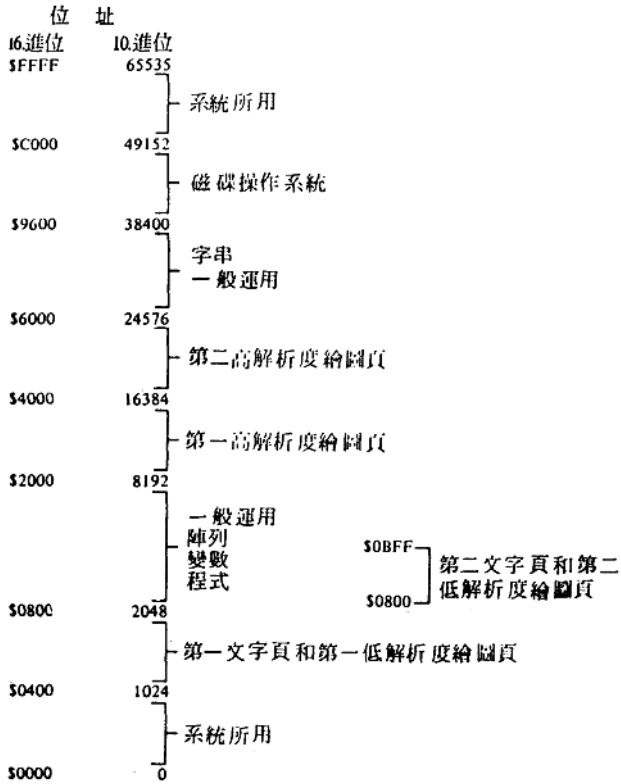


圖 4-2.1 APPLE II 記憶體的分配和應用

我們通常以十進位及十六進位表示位址。電腦本身是用二進位表示，但是這種表示法對我們人類而言，實在太過繁雜。

通常可將 65536 個記憶體以十進位編號爲 0 到 65535 （以十六進位表示爲 $0 到 $FFFF ）。當討論一個記憶體位置[

比如說$1B82（7042）]時，通常我們稱$1B（27）爲「高位元組」，而$82（130）爲「低位元組」（7042＝27×256＋130）。

$400（1024）到$7FF（2047）的記憶體有兩種用途：顯示正文和低解析度圖形。在 TEXT 模式中，電腦將這些記憶體的內容轉換成文字，而以 24 列，每列 40 個字的方式顯示出來；在GR 模式中，電腦將$400到 $7FF 記憶體的內容轉換爲圖形和文字，而在顯示幕下端顯示 4 列文字，上方顯示 40×40 的圖形矩陣。

$800（2048）到$BFF（3071）的記憶體也能顯示文字或圖形，但是Applesoft 程式通常佔據了這個區域。後面將會討論以 $800到$BFF記憶體顯示第二頁文字或圖形的技巧。

$2000（8192）到$3FFF記憶體是用來顯示第一高解析度繪圖頁。在 HGR 模式中，電腦顯示 280×160 的圖形矩陣，同時在顯示幕下方顯示文字。這4 列是第一頁文字（$400～$7FF）的最後4 列。

第二高解析度繪圖頁用 $4000（16384）到$5FFF（24575 ）記憶體。在 HGR2 模式中，電腦將這個區域的內容轉換成280 ×192 圖形矩陣，而不顯示任何文字。

在第二高解析度繪圖頁中，也可以在顯示幕下方顯示文字，這些文字來自第二文字頁（$800～$BFF）的最後 4 行。另外我們也可以在顯示第一高解析度繪圖頁時，不顯示出顯示幕下方的文字，在這種情形下就以 280×192的矩陣來顯示圖形。同樣的，第一和第二低解析度繪圖頁也都能以加上或去掉文字的方式顯示。

$400到$7FF 記憶體是用來顯示文字和第一低解析度繪圖頁，但是它的位址結構和你所想像的不一樣。圖 4-2.2說明當

這段記憶體用來做爲文字頁時，它的定址方式。

```
10  TEXT : HOME : VTAB 6: PRINT  TAB( 18);"TEXT"
20  PRINT  TAB( 15);"ADDRESSING"
```

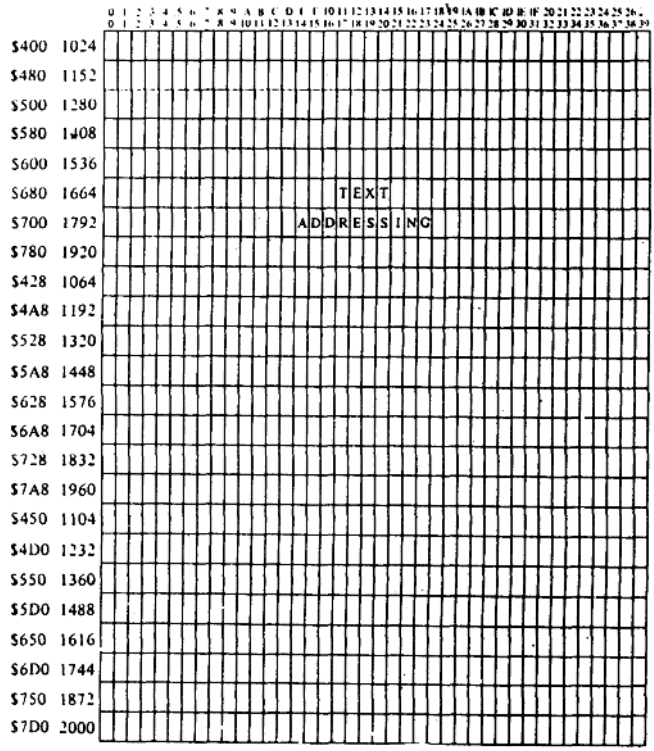


圖 4-2.2

4-3 一些基本模式的介紹

本書中所有的程式都是用Apple II Plus 48K 系統所寫的

，因此往後對於座標系統和繪圖指令的討論將與上述系統有關。然而，對於有繪圖能力的微電腦而言，操作是大同小異的。假如你沒有Apple微電腦，而有其他可繪圖的系統，我們所討論的仍可適用於你的系統，只是在將來發展程式時，必須做某種程度的修改。尤其是以下的三點可能必須更改：第一是座標軸的方向；第二是螢幕上標繪點（ plotting point ）的數目；第三是程式中的繪圖指令。我們將在接著的各段中繼續討論。

圖4-3.1是Apple II Plus的螢幕，X軸是在螢幕的上方從左延伸到右。Y軸是沿左側從上延伸到底下。

Apple II Plus有三種繪圖形式。第一種是低解析度（ Lowresolution ）繪圖，其影像是由較粗的方塊組成。在本書中將不使用這種方式。第二是高解析度（ high resolution ）繪圖第一頁（ page 1 ）。在此形式中，X的標繪值在0與279之間，Y可在0與159之間。換句話說，我們可以在以上的範圍中標繪點或是線。如果X或Y的值超出了規定範圍，這將會引起錯誤訊號（ error message ），並且做止程式的執行。在高解度第一頁時，有四行文字可顯示於繪圖區底下。這四行可用爲輸入指令或資料。第三種是高解度第二頁（ page 2 ）。在這形式中，X的值規定在0和279之間，這與第一頁是一樣的，但是Y可以從0到191。在這種形式時，螢幕底部的文字資料會被擦掉。同時，高解度第二頁要比第一頁更費記憶體，這會引起程式設計的問題。

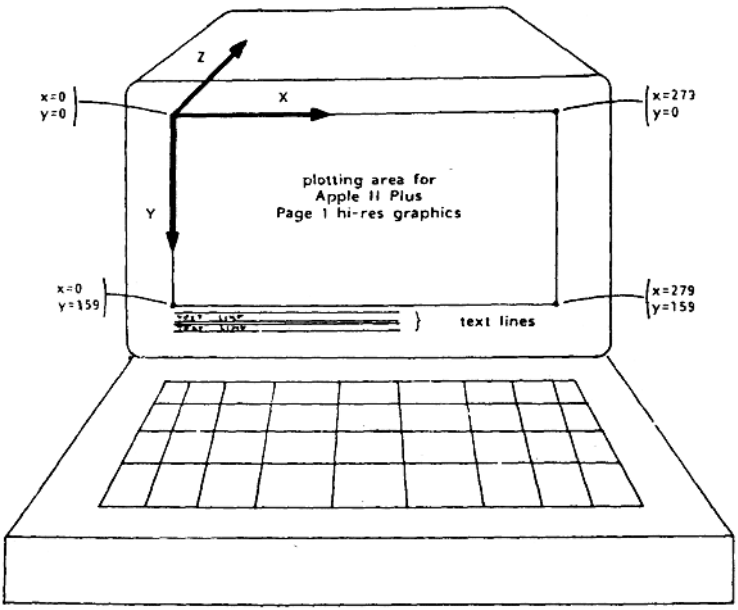


圖 4-3.1　APPLE II 的繪圖螢幕

(1)　TEXT模式

我們可以利用BASIC 裏的 POKE 指令來說明顯示幕的位址結構。 POKE 把一個數存入你所指定的記憶體位置中，例如 POKE 1930，25 把 25 存入1990 記憶體位置中。如果以 TEXT 模式顯示，POKE 指令將在 19 列第 31 個位置顯示出白底黑字的字母Y'，這是因爲它的碼以十進位表示是 25 。正文顯示幕的位址結構（圖 4-2.2）決定Y的位置。若想在白底黑字的Y的正上方顯示閃爍的Y（ASCII 字母碼爲 89），可用 POKE 1262，89 ；若想在白底黑字的Y的正下方顯示正常（

黑底白字）的Y，則用POKE 1518，217。

(2) GR模式

在GR模式中，記憶體內容的意義是完全不一樣的。顯示幕包含一個40×40的圖形矩陣和下方的4列文字。一個記憶體位置的內容控制兩個疊在一起的圖形方塊的顏色，上面方塊的顏色由低半位元組所含的數決定，下面方塊的顏色則由高半位元組所含的數決定。下面是一些說明它的方法的例子。

由於十進位25等於$19，POKE 1390，25將一個紫紅色（色碼爲1）的方塊放在橙色（色碼爲9）的方塊之下。1390記憶體控制位置（30，36）和（30，37）兩個方塊，使得（30，36）方塊爲橙色，而（30，37）方塊爲紅色。

第二個例子：我們知道十進位89等於$59，而且1262記憶體控制（30，34）和（30，35）兩個低解析度方塊，因此POKE 1262，89將（30，34）方塊塗成橙色，而（30，35）塗成灰色。同理，POKE 1518，217將把（30，38）方塊塗成橙色，把（30，39）方塊塗成黃色。

(3) HGR模式

在HGR或HGR2模式中，所顯示的影像是記憶體的一個個位元。我們可用定址方法以及POKE指令來說明這種圖形顯示的方法。

在HGR模式中，$2000（8192）記憶體位置控制顯示幕左上角的位置。電腦將8192記憶體位置的前7個位元顯示出來，那就是HGR顯示幕最上方一條線的前7個點；第一位元控制最左邊的點，第二位元控制第二個點，其餘由此類推。第8位元不顯示出來，成們用它來控制顏色。

舉個例子，設8192記憶體位置所含的數爲18（爲了確定

起見，請用下面的指令：POKE 8192，18 ），它以二進位表示為 00010010。由於圖形顯示以相反的次序把位元顯示出來，因此第二和第五點應該發亮，其餘則否。（見圖 4-3.2 ）

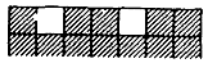


圖 4-3.2

＜例 1＞ 以 BASIC 程式寫一個位元樣式（ bit pattern ）的程式。

```
1  REM  PROGRAM      (BIT PATTERN)
2  REM  PLACES THE LETTER A ON THE HI RES SCREEN
10  DATA  8,20,34,34,62,34,34
20  HOME : HGR : VTAB 21
30  FOR I = 8192 TO 14366 STEP 1024
40  READ X
50  POKE I,X
60  INPUT K$
70  NEXT I
80  END
```

您每按一次 RETURN，這個字母的構成就多一個 byte

30 列開始的 FOR 迴圈會到 8192 、9216 、10240、11264 、12288 、13312 、14366 記憶體位置取資料，這些記憶體的內容將一個個顯示在顯示幕左上角。30 ～ 70 的迴圈執行完畢之後，這些記憶體所含的數如下：

記憶體位址	位元樣式（反向）	
	十進位	二進位
8192	8	0001000
9216	20	0010100
10240	34	0100010
11264	34	0100010
12288	62	0111110
13312	34	0100010
14366	34	0100010

顯　示

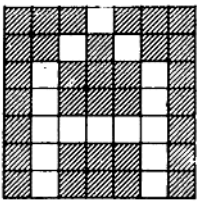


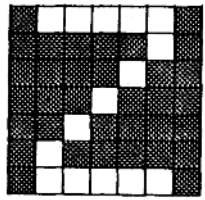
圖　4-3.3

當這些記憶體的內容顯示在顯示幕上時將出現字母「A」的影像。

＜例2＞　以BASIC程式寫一個位元樣式字母的程式

首先我們將字母或符號畫在7×7的格子上，如果想畫出字母「Z」，我們便依據圖4-3.4將每個位元樣式轉換成數碼，然後再將位元樣式 POKE 入記憶體中。

```
1  REM  PROGRAM      (BIT PATTERN CHARACTERS)
2  REM  PLACES LETTERS A AND Z ON THE HI RESCREEN
10  HOME : HGR : VTAB 21
20  DATA  8,20,34,34,62,34,34
30  FOR I = 8192 TO 14366 STEP 1024
40  READ X
50  POKE I,X
60  NEXT I
70  DATA  62,32,16, 8, 4, 2, 62
80  FOR I = 8193 TO 14367 STEP 1024
90  READ X
100  POKE I,X
110  NEXT I
120  END
```



顯示	位元樣式（反向）	
	二進位	十進位
	0111110	62
	0100000	32
	0010000	16
	0001000	8
	0000100	4
	0000010	2
	0111110	62

圖 4-3.4

4-4 如何以APPLEⅡ繪彩色的圖

在此不想深究高解析度顯示幕中彩色圖形的顯示特性，但是你必須知道一個重要的限制：並不是每個方塊都能塗上各種顏色。就以例題 1 的程式來說明這個事實：

＜例 1＞ 以 BASIC 程式繪出四條彩色垂直線

```
1  REM  PROGRAM     (COLOR PROBLEM)
2  REM  PLOTTED LINES ARE NOT VISIBLE
10  HOME : HGR : VTAB 21
20  HCOLOR= 2
30  FOR I = 2 TO 5
40  HPLOT 7 * I,10 TO 7 * I,30
50  NEXT I
60  END
```

請注意程式畫了 4 條垂直線，但是在顯示幕中，我們只看得見兩條，這是因爲 2 號顏色（藍色）只能畫在偶數行中。其他的顏色（除了白色和黑色之外）也有同樣的限制：1 號和 5 號顏色

只能畫在奇數行，2號和6號只能畫在偶數行，只有黑色和白色能顯示在每一行中。

爲了避免將有色的垂直線畫在它不能顯示出來的地方，可採用雙重畫法：

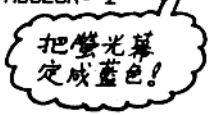
對　　HPLOT X,Y1 TO X,Y2

請加入下面的指令

```
HPLOT X+1,Y1 TO X+1,Y2
```

＜例2＞　本例是一個討論混合顏色的問題，請讀者們不妨將程式執行看看結果如何。

```
1  REM  PROGRAM      (PROBLEM WITH MIXED COLORS)
2  REM  PLOTTING OVEER BACKGROUND COLOR
10  HOME : VTAB 21: HGR : HCOLOR= 2
20  HPLOT 0,0
30  CALL 62454
40  HCOLOR= 5
50  HPLOT 1,1 TO 100,100
60  END
```



Apple所用的顏色編碼如下：

0——黑色1	4——黑色2
1——綠色（可變）	5——不定
2——藍色（可變）	6——不定
3——白色1	7——白色2

1，2，5，6號顏色爲不定色，它們由顯示幕監視器的彩度控制來決定。

在顯示幕形像上使用多種顏色時，你最好選擇0，1，2，3或是4，5，6，7這些顏色。在程式2.4中我們選擇藍色（

0行）做爲背景色（ 20 , 30行 ）。當我們以5號顏色畫出斜線（ 50 行 ）時，混色之後所產生的影像是好是壞端視觀測者的喜好而定。

字典的定義並不是都適用於日常生活，圖形顯示幕上的「直線」影像就是一個例子。理論上，直線的寬度（或是厚度）應該爲零。顯示直線的圖形用的是掃描點（ pixel ），它不是亮就是不亮。由於掃描點的大小不爲零，所以畫出來的直線的厚度並不爲零。而且掃描點是以行和列排列得很整齊，當我們畫斜線時，將會有很明顯的非線性效果。

舉個例子，請考慮4-4.1中的直線。如果我們使直線通過的掃描點發亮，它的形像很明顯的成階梯狀。
Applesoft中HPLOT X1,Y1 TO X2,X2 的指令會畫出如圖4-4.1所示的階梯狀直線。

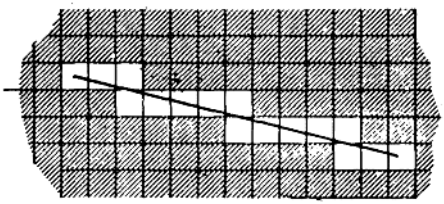


圖 4-4.1

定義直線的方式也會影響它的影像。當我們要畫一條直線時，Applesoft 必須決定那些掃描點應該發亮。爲了做計算，直線的方向十分重要。請試試下面的程式：

```
HGR: HCOLOR =3: HPLOT 10,10 TO 100,90
```

電腦會畫出一條斜線，請先不要清除顯示幕，再鍵入：